DS 574 Algorithmic Mechanism Design **Lecture #9**
Fall 2023 Prof. Kira Goldner

Linear Programming Duality

The Dual of a Linear Program

Every linear program has a *dual* linear program. We call the original linear program the *primal*. There are a bunch of amazing properties that come from LP duality.

Going back to our nutrition example, we want to find the dual linear program. A maximization problem's dual is a minimization problem. Here, we have a minimization problem, so the dual will be a maximization problem.

To take the dual: Label each primal constraint with a new dual variable. In our new linear program, each dual constraint will correspond to a primal variable. For the left-hand side, count up the appearances of this constraint's primal variable (e.g., x_1) in each of the primal constraints and multiply them by the dual variable for those constraints. That is, if x_1 appears 5 times ($5x_1$) in constraint for y_1, then add $5y_1$ to x_1's constraint. Don't forget to include its appearance in the primal's objective function, but this will be the right-hand side of the constraint. Finally, the dual objective function is given by the right-hand side coefficients and their correspondence to the dual variables via the constraints in the primal. (See below).

Primal:

$$\begin{aligned} \min \quad & 0.6y_1 + 0.35y_2 & & \\ \text{subject to} \quad & 5y_1 + 7y_2 \geq 8 & \text{(starch)} \quad & (x_1) \\ & 4y_1 + 2y_2 \geq 15 & \text{(proteins)} \quad & (x_2) \\ & 2y_1 + 1y_2 \geq 3 & \text{(vitamins)} \quad & (x_3) \\ & y_1, y_2 \geq 0 & \text{(non-negativity)} & \end{aligned}$$

Dual:

$$\begin{aligned} \max \quad & 8x_1 + 15x_2 + 3x_3 & & \\ \text{subject to} \quad & 5x_1 + 4x_2 + 2x_3 \leq 0.6 & \text{(grain 1)} \quad & (y_1) \\ & 7x_1 + 2x_2 + 1x_3 \leq 0.35 & \text{(grain 2)} \quad & (y_2) \\ & x_1, x_2, x_3 \geq 0 & \text{(non-negativity)} & \end{aligned}$$

Sometimes, the dual can even be interpreted as a related problem, as we will see.

The following is the normal form for a maximization problem primal and its dual:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned} \qquad \begin{aligned} \min \quad & \mathbf{y}^T\mathbf{b} \\ \text{subject to} \quad & \mathbf{A}^T\mathbf{y} \geq \mathbf{c} \\ & \mathbf{y} \geq 0 \end{aligned}$$

For the above example:

$$\mathbf{A} = \begin{bmatrix} 5 & 4 & 2 \\ 7 & 2 & 1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0.6 \\ 0.35 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 8 \\ 15 \\ 3 \end{bmatrix}$$

Example 3: Unweighted Maximum Matching

Given a graph $G = (V, E)$ choose a maximum size matching—a set of edges S such that no vertex is covered by more than one edge.

Decision variables: x_e indicating whether edge e is in the matching.

Primal Linear Program:

$$\begin{aligned} \max \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{e: v \in e} x_e \leq 1 && \forall v \quad \text{(vertex matched at most once)} \quad (y_v) \\ & x_e \geq 0 && \forall e \quad \text{(non-negativity)} \end{aligned}$$

Taking the dual of the above primal, we get the following linear program:

$$\begin{aligned} \min \quad & \sum_{v \in V} y_v \\ \text{subject to} \quad & \sum_{v \in e} y_v \geq 1 && \forall e \quad \text{(edge covered)} \quad (x_e) \\ & y_v \geq 0 && \forall v \quad \text{(non-negativity)} \end{aligned}$$

What problem is this? (Fractional) Vertex Cover!

Conditions for Optimality

Weak Duality

Theorem 1 (Weak Duality)**.** *If* $\mathbf{x}$ *is feasible in (P) and* $\mathbf{y}$ *is feasible in (D) then* $\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}$.

Proof.

$$\mathbf{c}^T\mathbf{x} \overset{1}{\leq} (\mathbf{A}^T\mathbf{y})\mathbf{x} = \mathbf{y}^T\mathbf{A}\mathbf{x} \overset{2}{\leq} \mathbf{y}^T\mathbf{b} = \mathbf{b}^T\mathbf{y}.$$

Where (1) follows by the dual constraints $\mathbf{A}^T\mathbf{y} \geq \mathbf{c}$ and (2) follows by the primal constraints $\mathbf{A}\mathbf{x} \leq \mathbf{b}$. □

This theorem says that *any* feasible solution to the primal is a *lower bound* to *any* feasible solution to the dual, and likewise, any feasible solution to the dual is an *upper bound* to the primal.

That is, fractional vertex cover gives an upper bound on how large the (fractional) maximum

matching can be, and likewise, fractional maximum matching gives a lower bound on how small the minimum (fractional) vertex cover can be.

What is not trivial (or by definition) is *strong duality*, and in fact, it is so involved that we will not even prove the hard direction: that an optimal solution always exists.

Conditions for Optimality

Strong Duality

Strong duality states that everything in fact needs to hold with equality to be optimal.

Theorem 2 (Strong Duality)**.** *A pair of solutions* $(\mathbf{x}^*, \mathbf{y}^*)$ *are optimal for the primal and dual respectively if and only if* $\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{y}^*$.

Proof. ($\Leftarrow$) The *if* direction is easy to see: we know that the dual gives an upper bound on the primal, so if these objectives are equal, then the primal objective that we are trying to maximize could not possible get any larger, as it's always *at most* the dual's objective. This is *as tight as possible*.

($\Rightarrow$) The *only if* direction is harder to prove, and we'll skip it for now. □

Complementary Slackness

We rewrite the primal and dual with each constraint separated, and then formalize another condition for optimality called *complementary slackness*, which states that for each corresponding constraint and variable, at most one can be slack in an optimal solution.

Primal (P):

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \sum_i a_{ji}x_i \le b_j \quad \forall j \quad (y_j) \\ & x_i \ge 0 \quad \forall i \end{aligned}$$

Dual (D):

$$\begin{aligned} \min \quad & \mathbf{y}^T\mathbf{b} \\ \text{subject to} \quad & \sum_i a_{ij}y_i \ge c_i \quad \forall i \quad (x_i) \\ & y_j \ge 0 \quad \forall j \end{aligned}$$

Theorem 3 (Complementary Slackness)**.** *A pair of solutions* $(\mathbf{x}^*, \mathbf{y}^*)$ *are optimal for the primal and dual respectively if and only if the following complementary slackness conditions (1) and (2) hold:*

$$\sum_i a_{ji}x_i = b_j \quad \textit{or} \quad y_j = 0 \qquad (1) \qquad\qquad \sum_i a_{ij}y_i = c_i \quad \textit{or} \quad x_i = 0. \qquad (2)$$

Proof. ($\Rightarrow$) According to complementary slackness, by rearranging our constraint, either $\sum_i a_{ji}x_i - b_j = 0$ or $y_j = 0$. This ensures that the multiplied quantity $\left(\sum_i a_{ji}x_i - b_j\right) y_j = 0$, as *one* of the two terms on the left-hand side must be 0. Then multiplying out and rearranging gives that $y_j \sum_i a_{ji}x_i = y_j b_j$. This process with all rows gives the equality from complementary slackness that $\mathbf{y}^T\mathbf{A}\mathbf{x} = \mathbf{y}^T\mathbf{b}$.

Similarly, using the condition that $\sum_i a_{ij}y_i = c_i$ or $x_i = 0$ gives that $\mathbf{c}^T\mathbf{x} = (\mathbf{A}^T\mathbf{y})\mathbf{x}$.

Then following our inequalities in the proof of weak duality, they now all hold with equality, so by Strong Duality, $(\mathbf{x}, \mathbf{y})$ are optimal solutions to the primal and dual.

$$\mathbf{c}^T\mathbf{x} \quad = \quad (\mathbf{A}^T\mathbf{y})\mathbf{x} \quad = \quad \mathbf{y}^T\mathbf{A}\mathbf{x} \quad = \quad \mathbf{y}^T\mathbf{b} \quad = \quad \mathbf{b}^T\mathbf{y}.$$

($\Leftarrow$) Similarly, if Strong Duality holds, the above inequalities hold with equality, in which case it must be that $y_j \sum_i a_{ji}x_i = y_j b_j$ for all j and $\sum_i a_{ij}y_i x_i = c_i x_i$ for all i, and hence that either $\sum_i a_{ji}x_i - b_j = 0$ or $y_j = 0$ for all j and that either $\sum_i a_{ij}y_i = c_i$ or $x_i = 0$ for all i. □

Maximizing Welfare in the Unit Demand Setting

Given n unit-demand bidders and m items, determine the allocation rule that maximizes welfare. We do this by formulating a linear program, determining our objective, decision variables, and constraints:

$$\begin{aligned}
\max \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}x_{ij} & & \\
\text{subject to} \quad & \sum_{i=1}^{n} x_{ij} \leq 1 & \forall j & \quad \text{(items allocated at most once)} \\
& \sum_{j=1}^{m} x_{ij} \leq 1 & \forall i & \quad \text{(bidders unit demand)} \\
& x_{ij} \geq 0 & \forall i, j & \quad \text{(non-negativity)}
\end{aligned}$$

We then formulate the dual:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} p_j & & \\
\text{subject to} \quad & u_i + p_j \geq v_{ij} & \forall (i,j) & \quad \text{(IC)} \\
& u_i, p_j \geq 0 & \forall i, j & \quad \text{(non-negativity)}
\end{aligned}$$

By rewriting our first constraint as

$$u_i \geq v_{ij} - p_j,$$

we reinterpret it as an incentive compatibility constraint. Instead of determining an allocation, we determine buyer utilities and item prices, the sum of which we minimize.

On our homework, we will use this to prove that for gross substitutes valuations, the optimal primal allocation $\mathbf{x}$ and dual solution $\mathbf{p}$ form a Walrasian equilibria, and this is precisely the valuation class for which welfare can be maximized in polynomial time.

We will need to use the following theory of when there exist polynomial-time algorithms for linear programs.

Separation Oracles

Fact 1 (Ellipsoid Algorithm)**.** Every linear program that admits a polynomial-time separation oracle can be solved in polynomial time.

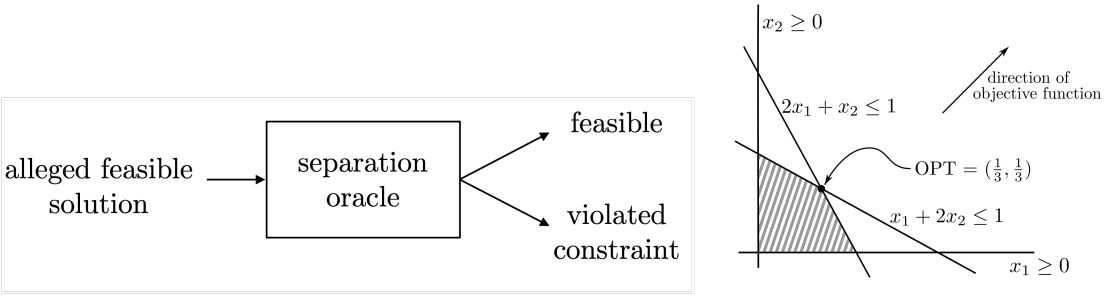

Figure 1: Left: A sketch of a separation oracle. For example, in the toy example on the right, on the alleged feasible solution $(\frac{1}{3}, \frac{1}{2})$, the separation oracle may return the violated constraint $x_1 + 2x_2 \leq 1$.

Consider a linear program such that:

- **a.** There are n decision variables.
- **b.** There are any number of constraints, for example, exponential in n. These constraints are not provided explicitly as input.
- **c.** There is a polynomial-time *separation oracle* for the set of constraints. By "polynomial-time," we mean running time polynomial in n and the maximum number of bits of precision required. A separation oracle (Figure 1) is a subroutine that takes as input an alleged feasible solution to the LP, and either (i) correctly declares the solution to be feasible, or (ii) correctly declares the solution to be infeasible, and more strongly provides a proof of infeasibility in the form of a constraint that the proposed solution violates.

(The ellipsoid algorithm is not actually practical, but there are other algorithms that *are* often practically useful that rely on a separation oracle, such as cutting plane methods.)